

Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
федерального государственного бюджетного образовательного учреждения высшего образования
**«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»**
(МИ ВлГУ)

Факультет информационных технологий
Кафедра информационных систем

КУРСОВАЯ РАБОТА

по курсу Управление данными
на тему: «Создание приложения для ведения учёта успеваемости»

Руководитель

К. Т. Н., доц. каф. ИС
(уч. степень, звание)

Симаков Р.А.
(фамилия, инициалы)

(подпись) (дата)

Студент ИС-119
(группа)

Королев Л.Я.
(фамилия, инициалы)

(подпись) (дата)

Члены комиссии

(подпись) (Ф.И.О.)

(подпись) (Ф.И.О.)

Муром, 2021

В курсовой работе отражено создание программы для ведения школьного журнала успеваемости учеников, на языке программирования python, с помощью фреймворка flask, проведены тестирование и отладка системы, приведены примеры работы программы. Данный проект может быть интересен всем, кто интересуется работой с базами данных.

Табл. 1. Ил. 1. Библ. 1.

Содержание

Ведение.....	5
1 Анализ технического задания.....	6
1.1 Требования к функциональным характеристикам.....	6
1.1.1 Пользователь.....	6
1.1.2 Директор.....	6
1.1.3 Учитель.....	10
1.1.4 Ученик.....	10
2 Разработка БД.....	11
3 Разработка программы.....	16
3.1 Начало разработки.....	16
3.2 Создание страницы ученика.....	19
3.3 Создание страницы учителя.....	21
3.4 Создание страницы администратора.....	23
4 Тестирование.....	27
Заключение.....	30
Список литературы.....	31

					МИВУ 09.03.02 - 00.000 ПЗ						
Изм.	Лист	№ докум.	Подп.	Дата	<u>Создание приложения для ведения учёта успеваемости</u>			Лит.	Лист	Листов	
Разраб.								у		4	27
Пров.								МИ ВлГУ ИС-119			
Н. контр.											
Утв.											

Введение

С развитием повсеместного всеобщего образования, появилась необходимость следить за качеством получаемых людьми знаний, для этого в школах по сей день используется пятибалльная система оценивания. Для ведения учета успеваемости каждого из многочисленных учеников используются классные журналы, содержащие в себе информацию о успеваемости каждого ученика конкретного класса по данному предмету.

На протяжении долгого времени использовались исключительно бумажные варианты данного документа, что включает в себе массу проблем: большое количество используемой бумаги, как следствие масса мукулатуры, высокая пожароопасность в архивных помещениях. Также бумажный вариант документа будет безвозвратно утерян в случае воздействия внешних сил.

Тема данной курсовой работы – разработка программы для ведения подобного журнала в электронном виде, с использованием базы данных.

Целью курсовой работы является изучение и закрепление принципов работы с базами данных на примере разработки программы для ведения школьного журнала.

Исходя из цели, определены следующие задачи:

1. Изучение материалов по теме работы
2. Разработка базы данных;
3. Выбор технологии разработки программы;
4. Разработка интерфейса;
5. Тестирование и отладка программы.

Результатом данной курсовой работы будет являться программа, в которой будут реализованы все поставленные задачи.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						5
Изм.	Лист	№ докум.	Подп.	Дата		

1. Анализ технического задания

1.1 Требования к функциональным характеристикам:

По заданию, требуется создать программу, в которой реализована возможность ведения школьного журнала.

В ходе работы планируется создать сайт, позволяющий пользователям быстро и просто получать доступ к спискам успеваемости, для их просмотра или редактирования. Для разработки системы был выбран фреймворк Flask в совокупности с СУБД Red Database.

Для наиболее корректного понимания пути разработки, был создан перечень ситуаций использования (use cases).

1.1.1 Пользователь

- заходит на сайт
- открывается форма авторизации
- в поле ввода логина вводится логин
- в поле ввода пароля вводится пароль
- Нажимает кнопку авторизации
- происходит авторизация
- пользователя перенаправляет на страницу, соответствующую его учетной записи

Данный процесс выделен в отдельную группу, так как пользователь любого уровня должен пройти процедуру авторизации перед тем, как попасть в какой-либо раздел.

1.1.2 Директор (Зав. Учитель)

Директор добавляет новый класс:

- авторизуется
- нажимает кнопку "Классы"
- нажимает кнопку "Добавить"
- открывается страница создания класса

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подп.	Дата		

- вводит информацию о классе в поля
- выбирает номер и литеру класса
- нажимает кнопку подтверждения
- класс добавляется в общий список

Директор удаляет класс:

- авторизуется
- нажимает кнопку "Классы"
- нажимает на нужный класс
- открывается страница класса
- нажимает кнопку удаления
- класс удаляется из общего списка

Классы будут иметь свой индивидуальный код, например, комбинация из года поступления и порядкового номера. Отдельно будут храниться номера классов с литерами. Так как для большинства учебных заведений программа составляется одинаковая, и не претерпевает значимых изменений, база данных будет сразу поставляться с готовым набором номеров классов с прописанной программой. При надобности данные параметры можно будет редактировать.

Директор добавляет ученика:

- авторизуется
- нажимает кнопку Ученики
- открывается страница со списком учеников
- нажимает кнопку добавления
- открывается окно добавления ученика
- Заполняет данные о ученике (ФИО, и т.д.)
- Нажимает кнопку выбора класса
- выбирает класс, в который зачислен ученик

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подп.	Дата		

- нажимает кнопку добавления
- ученик добавлен в общий список и привязан к классу

При добавлении ученика не обязательно указывать класс. Это можно сделать позже. Данная возможность предназначена для учеников, пребывающих в учебный процесс по мере его течения.

Директор добавляет Учителя:

- авторизуется
- нажимает кнопку Учителя
- открывается страница со списком учителей
- нажимает кнопку добавления
- открывается окно добавления учителя
- Заполняет данные об учителе (ФИО)
- нажимает кнопку добавления
- учитель добавлен в общий список

В один класс может быть добавлено несколько учителей по одному предмету, например, для уроков труда, где учитель определяется полом ученика.

Директор редактирует данные программы класса:

- авторизуется
- нажимает кнопку классы
- открывается страница со списком классов
- нажимает на кнопку редактировать
- открывается окно выбора номера класса
- нажимает на нужный класс
- открывается форма редактирования программы номера класса

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подп.	Дата		

- нажимает кнопку добавить предмет (или нажимает на предмет, если уже был добавлен)
- открывается форма составления плана
- заполняются данные о количестве часов
- заполняются данные о преподавателях, ведущих предмет
- нажимает кнопку подтверждения
- информация о количестве часов добавлена
- для данного класса и предмета создается таблица успеваемости/посещаемости учеников

Директор добавляет предмет:

- авторизуется
- нажимает кнопку предметы
- открывается страница со списком предметов
- нажимает кнопку добавления
- открывается окно добавления предмета
- Заполняет данные о предмете (название)
- нажимает кнопку добавления
- предмет добавлен в общий список предметов

Данные 2 опции будут необходимы только в случае узкой специализации школы, например технический или гуманитарный уклон, что влечет появление дополнительных предметов, или увеличение количества часов в программе у имеющихся.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подп.	Дата		

1.1.3 Учитель

Учитель делает запись о проведенном уроке

- авторизуется
- выбирает нужный класс из раздела необходимого предмета
- открывается таблица журнала
- нажимает на ячейку таблицы, в которую необходимо выставить оценку, или изменить существующую

- открывается страница выставления оценки.
- заполняет данные об оценке
- нажимает кнопку подтверждения

1.1.4 Ученик (Родитель ученика)

- Ученик смотрит на свою успеваемость
- авторизуется
- открывается страница с таблицей успеваемости/посещаемости по каждому предмету.

Главная задача данной работы – обеспечить удобное взаимодействие учителя и ученика, или его родителей. Планируется реализовать следующие возможности: Возможность выставления не только оценки, но и отметки посещаемости. Также к конкретной оценке можно добавить комментарий, например, «2 (Не выполнил домашнее задание)», или «5 (Ответ у доски)». Родитель или ученик же должны видеть динамику выставления оценок, для чего будет сделана система отображения новых событий, произошедших после выхода из системы.

Также, для безопасности системы, у разных ролей пользователей будут различные привилегии для работы с системой.

Директор – может полностью управлять базой данных. Вносить изменения в любые таблицы. Учитель ограничен редактированием таблицы успеваемости только у тех классов, и по тем предметам, которые он ведёт. Ученик не может вносить правки в какие-либо данные таблицы, он может только просматривать данные о своей успеваемости.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						10
Изм.	Лист	№ докум.	Подп.	Дата		

2. Разработка БД

Для создания базы данных сначала необходимо определить, с какими предметами в данной ситуации мы имеем дело в реальной жизни. В школе учитель ставить оценки ученикам, по определенному предмету, на определенную дату. Рассмотрим объекты, которые встречаются в данной ситуации.

Первый объект – учитель. Учитель ведёт конкретный предмет у конкретного класса. У учителя есть ФИО, привязка к предмету, привязка к классу, у которого он ведёт предмет.

Второй объект – класс. Можно сказать, что класс – сущность абстрактная. Будем так считать, так как класс в школе – группа людей. Но каждый год у данной группы меняется номер. Поэтому класс можно разбить на 2 сущности: класс, как абстрактная сущность, и группа, как набор учеников.

Третий объект – ученик. Ученик принадлежит классу, он получает оценки по разным предметам в определенную дату. У него тоже есть ФИО, как и у учителя.

Четвертый объект – сам предмет. Предмет может быть один, но к нему могут принадлежать разные учителя и классы. Для этого потребуется создать связующую сущность в дальнейшем.

Пятый объект – оценка. Оценка ставится конкретным учителем, по конкретному предмету, конкретному ученику, на конкретную дату.

Для реализации данной задачи необходимо создать базу данных, хранящую в себе все необходимые данные. Исходя из задачи, были выявлены следующие необходимые сущности:

Список Учителей. Данная таблица должна содержать в себе основную информацию об учителе, а именно: фамилию, имя и отчество, и первичный ключ ID, для связи с другими сущностями. В данную таблицу не будет добавлено информации о предмете, так как для связи с предметом планируется создание отдельной таблицы.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подп.	Дата		

TEACHER		
ID	INTEGER	(PK)
FIRST_NAME	VARCHAR(255)	
SECOND_NAME	VARCHAR(255)	
FATHER_NAME	VARCHAR(255)	

Рис1.1 – Таблица «Учитель»

Список Предметов. Эта таблица планируется по аналогии со страницей учителя. Содержит в себе название и первичный ключ предмета.

SUBJECT		
ID	INTEGER	(PK)
NAME	VARCHAR(255)	

Рис1.2 – Таблица «Предмет»

Список классов. Так как каждый группы учеников меняют свой класс, было принято решение обособить сущность классов от сущности групп учеников. Группа класса содержит в себе простую информацию о себе, такую как литера класса, номер, и первичный ключ. В дальнейшем эта таблица будет использована для связи предметов, учителей и группы учеников.

CLASS		
ID	INTEGER	(PK)
NUMBER	INTEGER	
LITERA	VARCHAR(64)	

Рис1.3 – Таблица «Класс»

Список групп. Данная сущность будет содержать в себе собственный первичный ключ, внешний ключ, ссылающийся на класс, и дату поступления группы. Данная таблица необходима для объединения учеников, и облегчения их перевода в другой класс.

CLASS_GROUP		
ID	INTEGER	(PK)
START_DATE	DATE	
CLSS_ID	INTEGER	(FK)

Рис1.4 – Таблица «Группа»

Список учеников. Данная сущность совмещает свойства сущностей учителя и класса. Содержит в себе информацию о ФИО ученика, первичный ключ, и внешний ключ, ссылающийся на группу учеников.

STUDENT		
ID	INTEGER	(PK)
GROUP_ID	INTEGER	(FK)
FIRST_NAME	VARCHAR(255)	
SECOND_NAME	VARCHAR(255)	
FATHER_NAME	VARCHAR(255)	

Рис1.5 – Таблица «Ученик»

Сущность «План» . Содержит в себе внешние ключи на класс, учителя и предмет. Необходима для связи всех перечисленных сущностей. Также содержит первичный ключ, для однозначного определения записи среди других.

PLAN		
TEACHER_ID	INTEGER	(FK)
SUBJECT_ID	INTEGER	(FK)
CLASS_ID	INTEGER	(FK)
ID	INTEGER	(PK)

Рис1.6 – Таблица «План»

Список оценок. Данная сущность будет наиболее важной, и наиболее ёмкой в данной базе. Для однозначной идентификации принадлежности оценки к ученику и предмету, в таблице хранятся внешние ключи на студента и запись в таблице «план». Также, для расставления оценок в определенном порядке в интерфейсе программы, оценке присваивается дата выставления. Также в таблице хранится сама оценка, тип оценки, и комментарий к ней.

MARK		
STUD_ID	INTEGER	(FK)
PLAN_ID	INTEGER	(FK)
DATE_MARK	TIMESTAMP	
MARK	INTEGER	
TYPE_MARK	INTEGER	
COMMENT_FOR_MARK	VARCHAR(1000)	
ID	INTEGER	(PK)

Рис1.7 – Таблица «Оценка»

1) Таблица пользователей. Данная таблица содержит в себе список пользователей программой. Хранит в себе логин, пароль(его хеш), роль пользователя, и ID пользователя. Роль и ID нужны для определения, кем является пользователь, чтобы осуществить переадресацию после авторизации на нужную страницу сайта. Данная сущность не имеет связи с другими таблицами, т.к. внешний ключ не может ссылаться на несколько таблиц одновременно, а учителя и ученики разведены в разные сущности.

USERS		
ID	INTEGER	(PK)
USERNAME	VARCHAR(255)	
PASSWORD	VARCHAR(255)	
ROLE	INTEGER	
USER_ID	INTEGER	

Рис1.8 – Таблица «Пользователь»

Возможно, в процессе выполнения работы будут внесены изменения в базу данных.

Для хранения данных была создана база данных, ER-диаграмма которой приведена ниже:

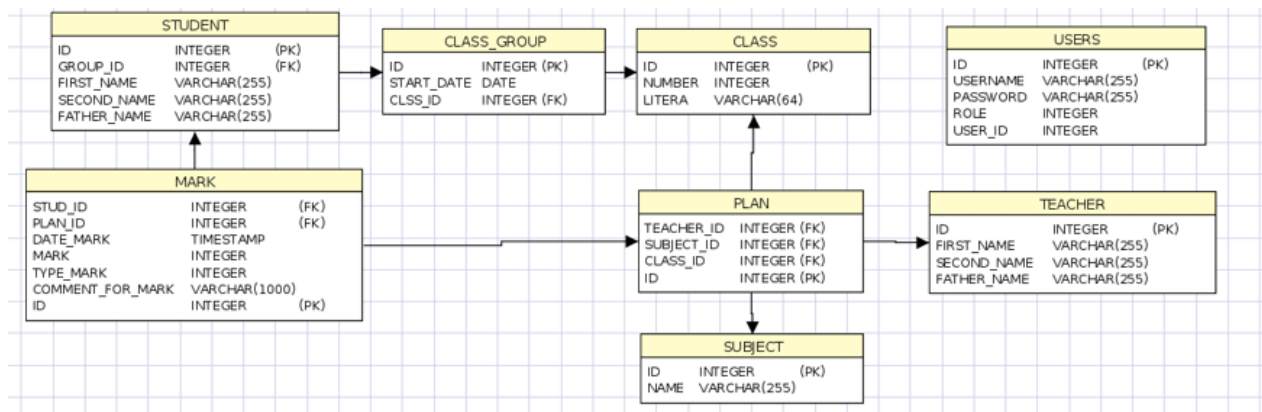


Рисунок 2 – ER-диаграмма базы данных

3. Разработка приложения

3.1 Начало разработки

Для того, чтобы разработать приложение, необходимо определить, на основе чего будет осуществляться разработка. Так как электронный журнал – приложение, к которому пользователь должен получать быстрый и простой доступ, было принято решение о создании сайта.

Для реализации этой задачи был выбран python фреймворк Flask, в совокупности с СУБД Red Database, работающая в портативном режиме с данным фреймворком.

Перед началом непосредственно самой разработки приложения, необходимо составить прототип интерфейса программы, для ускорения разработки.

Для создания прототипа интерфейса приложения был разработан прототип интерфейса в графическом редакторе Figma. Возможно, итоговый интерфейс будет отличаться от представленного.

Электронный журнал

Индификатор

Пароль


Авторизация

Рис3.1 – Прототип страницы авторизации

Электронный журнал

Оценки !

Ученик

USERNAME 


Предмет \ Дата	«ТУТ ДАТЫ»						История
П Р Е Д М Е Т Ы	5						
	3						
Комментарий							

Рис3.2 – Прототип страницы просмотра оценок учеником

Электронный журнал

Оценки !

Ученик

USERNAME 

Иванов Иван Иванович

Класс: 11

Ваши предметы:

- Литература
- Русский язык
- Иностранный язык
- Алгебра
- Геометрия
- Физика
- Информатика
- Физкультура
- ОВЖ

Вам стоит обратить внимание на предметы:

- Литература
Причина: низкий средний балл - 3
- Иностранный язык
Причина: пропуски без уважительной причины

Рис3.3 – Прототип страницы просмотра информации об ученике

Рис3.4 – Прототип страницы выбора предмета учителем

Рис3.5 – Прототип страницы выставления оценки учителем

После создания прототипа интерфейса, можно приступать к разработке самой программы.

3.2 Создание страницы авторизации

Было принято решение начать разработку со страницы авторизации. Страница регистрации в данном проекте отсутствует, так как эту функцию возьмет на себя администратор школы, чья учетная запись будет сразу занесена в базу данных.

Для осуществления различения пользователей, в базе данных была внесена роль пользователя, которая в дальнейшем будет проверяться в приложении для выбора из базы данных нужной информации.

Процесс авторизации происходит следующим образом: Пользователь вводит свой логин и пароль, нажимает кнопку «войти». Приложение принимает данные из формы браузера, и ищет подходящий логин в базе данных. Если такой логин существует, пароль из формы хэшируется и сравнивается с хэшем из базы данных. Если и пароль введен верно, программа проверяет роль пользователя, и загружает соответствующие данные из базы данных. В случае неверного ввода пароля, пользователю выводится соответствующее сообщение.

После загрузки данных происходит редирект на базовую страницу пользователя. У ученика это будет страница с оценками, у учителя страница со списком предметов, у администратора список таблиц, доступных для редактирования.

Ниже представлена функция, осуществляющая авторизацию пользователя:

```
@bp.route('/login', methods=('GET', 'POST'))
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        db = get_db()
        error = None
        user = db.cursor().execute('SELECT * FROM users WHERE username = ?',
        (username,)).fetchonemap()
        type = 0
        if user is None:
            error = 'Неверное имя пользователя.'
        elif not check_password_hash(user['password'], password):
            error = 'Неверный пароль.'

        if error is None:
            session.clear()
            session['user_id'] = user['id']
            session['user_type'] = user['role']
```

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						19
Изм.	Лист	№ докум.	Подп.	Дата		

```

        session['id'] = user['user_id']
        if user['role'] == 1:
            return redirect(url_for('student.about', mark = 0))
        if user['role'] == 2:
            return redirect(url_for('teacher.subs'))
        if user['role'] == 3:
            return redirect(url_for('admin.main'))
    flash(error)

```

```

    return render_template('auth/login.html')

```

Функция, загружающая подходящую информацию:

```

@bp.before_app_request
def load_logged_in_user():
    user_id = session.get('user_id')
    user_type = session.get('user_type')
    id = session.get('id')
    if user_type == 1:
        if user_id is None:
            g.user = None
        else:
            g.user = get_db().cursor().execute('SELECT * FROM student WHERE id = ?',
(id,)).fetchonemap()
    if user_type == 2:
        if user_id is None:
            g.user = None
        else:
            g.user = get_db().cursor().execute('SELECT * FROM teacher WHERE id = ?',
(id,)).fetchonemap()
    if user_type == 3:
        if user_id is None:
            g.user = None
        else:
            g.user = get_db().cursor().execute('SELECT * FROM users WHERE id = ?',
(user_id,)).fetchonemap()

```

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						20
Изм.	Лист	№ докум.	Подп.	Дата		

3.3 Создание страницы ученика

После создания страницы авторизации, начата разработка со страницы ученика, так как она является самой простой и быстро реализуемой из всех. Для вывода оценок на страницу браузера использована html таблица. В самой программе из базы данных загружается информация об ученике, о его предметах и об оценках. Также, исходя из текущей даты создается список дат текущей или ближайшей четверти, исключая выходные дни.

Для отображения информации, по конкретной оценке, функция принимает значения первичного ключа оценки, и загружает информацию о ней. Если такой оценки нет, тогда данный элемент шаблона отсутствует.

Также была добавлена возможность просмотра истории оценок по дате их выставления. Для этого информация об оценках сортируется по дате.

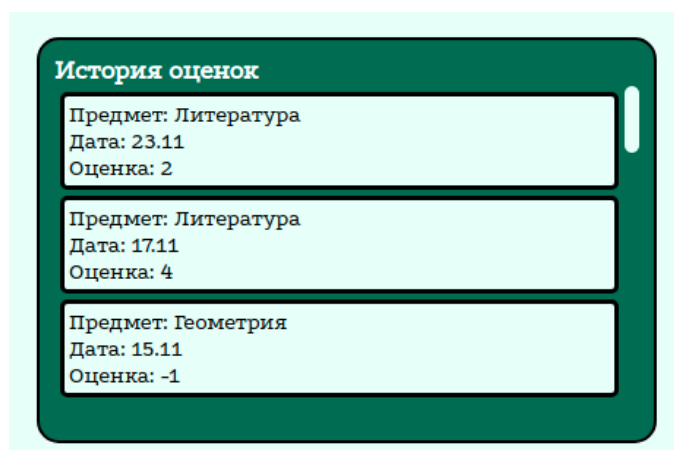


Рис4.1 – История оценок

Таким образом, ученик, попав на страницу, увидит таблицу оценок, историю оценок, и информацию по выбранной оценке.

Эта таблица заполняется путём перебора. Сначала перебираться все предметы, внутри каждого предмета перебираться даты. Внутри каждой даты перебираться все оценки. Если оценка принадлежит этой дате, и этому предмету, она выставляется в текущую ячейку. На ячейке создается метка, содержащая первичный ключ оценки, позволяющий загрузить информацию о ней в элемент просмотра информации. Для удобного просмотра содержимого у таблицы присутствует ползунок для горизонтального скроллинга.

При нажатии на ячейку с оценкой страница обновляется с загруженной информацией об оценке. Возможно, это не самое оптимальное решение, но во время разработки было принято решение использовать именно этот вариант, как наиболее простой и доступный в использовании.

Рис4.2 – Информация об оценке

3.3 Создание страницы Учителя

Часть программы, предназначенная для учителя, состоит из 2-х частей. Первая часть – страница выбора классов, у которых по его предметам учитель ведёт занятия. Вторая - таблица оценок по выбранному предмету у выбранного класса.

Первая страница создается просто – из базы данных выбираются записи из таблицы ПЛАН, где ID учителя соответствует ID текущего пользователя. Далее, из этих записей получается информация о предметах, которые ведёт учитель, и классы, которые учатся у данного учителя по данному предмету. Всё это представлено в несложном двухуровневом списке. На кнопки с обозначением класса создается ссылка на функцию, которая вызывает шаблон с таблицей для выставления оценок.

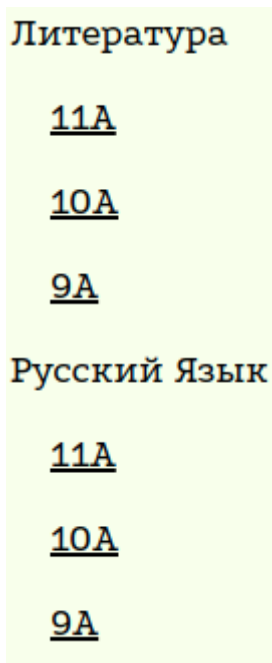


Рис4.3 – список классов

Вторая страница похожа на страницу ученика – она представляет собой страницу с таблицей, но не содержит полей истории оценок и информации об оценке. При нажатии на ячейку таблицы, если она пустая – откроется форма выставления оценки. Если в ячейке уже стоит оценка – форма редактирования оценки. Данные формы – дополнительные страницы для раздела учитель. Они

практически не отличаются друг от друга, единственная разница – при создании оценки вызывается SQL функция INSERT, а при обновлении SQL функция UPDATE.

Запрос к БД для вставки оценки:

```
db.cursor().execute(
    'INSERT into MARK (MARK.STUD_ID, MARK.PLAN_ID, MARK.DATE_MARK,
    MARK.MARK, MARK.TYPE_MARK, MARK.COMMENT_FOR_MARK) '
    ' values (?, ?, ?, ?, ?, ?) ',
    (student_id, plan_id, date, mark[0], mark_type, comment)
)
db.commit()
```

Запрос к БД для редактирования оценки:

```
db.cursor().execute(
    'UPDATE MARK '
    ' SET MARK.MARK = ?, MARK.COMMENT_FOR_MARK = ? '
    ' WHERE MARK.ID = ? '
    ,
    (mark[0], comment, mark_id)
)
db.commit()
```

Заполнение таблицы происходит по похожей со страницей ученика схеме. Только список предметов заменён на список учеников.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						24
Изм.	Лист	№ докум.	Подп.	Дата		

3.4 Создание страницы администратора

Страница администратора при использовании приложения будет самой незаметной, так как редактирование, удаление и добавление полей требуется только при внедрении системы, и при необходимости редактирования базы данных. Главная страница представляет собой список сущностей базы данных, а именно «Ученики», «Учителя», «Классы», «Группы», «Предметы», «План», «Пользователи». Каждая кнопка ведёт на страницу, в которой в виде таблицы отображается список объектов данной сущности. Для каждой таблицы был создан CRUD-пакет, позволяющий почти полностью управлять базой данных.

Для таблиц «Учителя», «Предметы» созданы простейшие формы создания и редактирования, так как эти таблицы не ссылаются ни на одну из других таблиц. Для таблицы «Учителя» сделано три поля ввода – Имя, Фамилия, Отчество. Для таблицы «Предметы» – Только название предмета. Первичный ключ автоматически создается средствами СУБД.

Для Таблицы «Классы» создана похожая форма, но для ввода номера класса поле ограничено значениями от 1 до 11 средствами языка вёрстки HTML.

Форма для создания и редактирования таблиц «Группы» и «План» имеет более сложную структуру, так как для группы необходимо выбирать класс, к которой она будет принадлежать. Для этого из базы данных выбирается список всех доступных классов, и загружается в элемент на форме. Если администратор выберет класс, уже занятый другой группой, программа выведет на экран сообщение об ошибке. В таблице «План» действия происходят аналогичным образом, но на этой странице присутствуют поля, соответствующие данной таблице.

Раздел пользователи имеет сложность – пользователям необходимо иметь пароль и логин для входа. При добавлении пользователя пароль генерируется случайным образом, и в базу записывается его хэш. Выходит, что вывести пароль каждого пользователя на постоянной основе не получится, так как хэш – шифрование одностороннее, и из хэша не может быть вычислен пароль. Для

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						25
Изм.	Лист	№ докум.	Подп.	Дата		

преодолена данной проблемы при создании пользователя администратора направляет на особую страницу, в которую выводится данные о пользователе с ещё не захешированным паролем. Данная страница отображается один раз, и не может быть воспроизведена снова. В дальнейшем администратор может сменить пароль пользователя, если тот его забудет.

Новые пользователи

ВНИМАНИЕ! ДАННАЯ СТРАНИЦА ОТОБРАЖАЕТСЯ ОДИН РАЗ.

Имя	Фамилия	Отчество	Username	Пароль	
Пётр	Убегаев	Владимирович	PetrUV1008	x5bsDy	Учитель

Рис4.4 – Страница список новых пользователей

4. Тестирование

Одним из важнейших этапов создания программного продукта является его тестирование и отладка. Тестирование позволяет выявить скрытые и явные недостатки программы, либо убедиться в ее пригодности для применения.

Целью тестирования является проверка работоспособности программы, т.е. правильности выполнения всех функций, описанных выше.

Для проверки функциональности программы проведены следующие тесты

Таблица 1 –Тестирование программы

№	Действие	Результат
1	2	3
1	Авторизация ученика	Успех
2	Авторизация учителя	Успех
3	Авторизация администратора	Успех
4	Выход из учетной записи ученика	Успех
5	Выход из учетной записи ученика	Успех
6	Выход из учетной записи ученика	Успех
7	Загрузка страницы ученика	Успех
8	Просмотр истории оценок	Успех
9	Просмотр информации об оценке	Успех
10	Вывод информации об предметах и классах учителя	Успех
11	Выбор класса по предмету	Успех
12	Вывод таблицы оценок	Успех
13	Выбор ячейки без оценки	Успех
14	Вывод формы для выставления оценки	Успех
15	Выставление оценки	Успех
16	Выбор ячейки с оценкой	Успех
17	Вывод формы редактирования оценки	Успех

18	Редактирование оценки	Успех
19	Выбор формы редактирования таблицы учеников	Успех
20	Удаление ученика	Успех
21	Вывод формы добавления ученика	Успех
22	Добавление ученика	Успех
23	Вывод формы редактирования ученика	Успех
24	Редактирование ученика	Успех
25	Выбор формы редактирования таблицы учителей	Успех
26	Удаление учителей	Успех
27	Вывод формы добавления учителей	Успех
28	Добавление учителей	Успех
29	Вывод формы редактирования учителей	Успех
30	Редактирование учителей	Успех
31	Выбор формы редактирования таблицы предметов	Успех
32	Удаление предметов	Успех
33	Вывод формы добавления предметов	Успех
34	Добавление предметов	Успех
35	Вывод формы редактирования предметов	Успех
36	Редактирование предметов	Успех
37	Выбор формы редактирования таблицы плана	Успех
38	Удаление плана	Успех
39	Вывод формы добавления плана	Успех
40	Добавление плана	Успех
41	Вывод формы редактирования плана	Успех
42	Редактирование плана	Успех

43	Выбор формы редактирования таблицы группы	Успех
44	Удаление группы	Успех
45	Вывод формы добавления группы	Успех
46	Добавление группы	Успех
47	Вывод формы редактирования группы	Успех
48	Редактирование группы	Успех
49	Выбор формы редактирования таблицы класса	Успех
50	Удаление класса	Успех
51	Вывод формы добавления класса	Успех
52	класса	Успех
53	Вывод формы редактирования класса	Успех
54	Редактирование класса	Успех

По результатам тестирования выявлено, что программа не имеет логических ошибок.

Данная программа тестировалась на компьютере следующей конфигурации:

1. Процессор: Intel Core i5-4210H 2.9GHz;
2. ОЗУ: 16.00 Гб;
3. ОС: Red OS 7.3 64-разрядная.

Заключение

В данной курсовой работе были расширены знания, полученные в ходе лекционных и практических занятий. Получены навыки самостоятельной работы по формализации поставленной задачи, программированию, тестированию и отладке. Разработана программа для ведения учёта успеваемости в общеобразовательных школах.

Данная программа является инструментом для ведения учёта успеваемости, который может быть использован в различных общеобразовательных учреждениях.

В дальнейшем возможно усовершенствование программы по функциональным и оптимизационным параметрам, путем улучшения безопасности приложения, добавлением статистики и истории оценок за предыдущие периоды времени, возможностью выставления четвертных и годовых оценок, созданием системы уведомлений о новых и измененных оценках, улучшении и доработки интерфейса всех страниц сайта.

В ходе курсовой работы были приобретены навыки работы с базами данных. Была решена задача выбора технологии разработки структуры базы данных и самой программы. Было осуществлено тестирование и отладка программы.

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						30
Изм.	Лист	№ докум.	Подп.	Дата		

Список литературы

1. Лекции по дисциплине УД.
2. Разработка web-приложение на Flask // URL: <https://romansimakov-reddatabaselab.readthedocs.io/ru/latest/flaskr.html> (дата обращения 16.11.21)
3. База знаний по HTML и CSS // URL: <https://htmlbook.ru/> (дата обращения 17.11.21)
4. О дате и времени в python // URL: <https://webformymself.com/data-i-vremya-v-python/> (дата обращения 17.11.21)
5. Простой генератор паролей на python // URL: <https://gist.github.com/ildarkhasanshin/aec24c301e9a2aaf5830444ccab0a4e5> (дата обращения 1.12.21)

					МИВУ 09.03.02 – 07.000 ПЗ	Лист
						31
Изм.	Лист	№ докум.	Подп.	Дата		